(12) UK Patent Application (19) GB (11) 2 273 418 (13) A

(43) Date of A Publication 15.06.1994

(21) Application No 9323325.2

(22) Date of Filing 11.11.1993

(30) Priority Data
(31) 975353 (32) 12.11.1992 (33) US

(71) Applicant(s)
Rockwell International Corporation

(Incorporated in USA - Illinois)

1431 Opus Place, PO Box 1494, Downers Grove, Illinois 60515, United States of America

(72) Inventor(s)
Joseph E Bloom
Susan A Palermo
Michael C Auclair

(51) INT CL[5]
H04M 3/42 , H04Q 3/545

(52) UK CL (Edition M)
H4K KFD

(56) Documents Cited
WO 92/11724 A1 WO 92/11603 A1 US 5117372 A
US 4695977 A

(58) Field of Search
UK CL (Edition M) **H4K KFD KFH**
INT CL[5] **H04M 3/42 3/50 3/54 , H04Q 3/00 3/545**
ONLINE DATABASES : WPI

(74) Agent and/or Address for Service
Shaw, Bowker & Folkes
Whitehall Chambers, 23 Colmore Row, BIRMINGHAM, B3 2BL, United Kingdom

(54) Programming a telephone system

(57) In order to make the programming of a telephone system, (for example an A.C.D. system), more user-friendly, a programming device provides the user with increased flexibility by enabling on-site programming. Script sets which define routing vectors are edited or manipulated using a high level language in order to customise the routing of incoming telephone calls in a manner as desired at the site of the user.

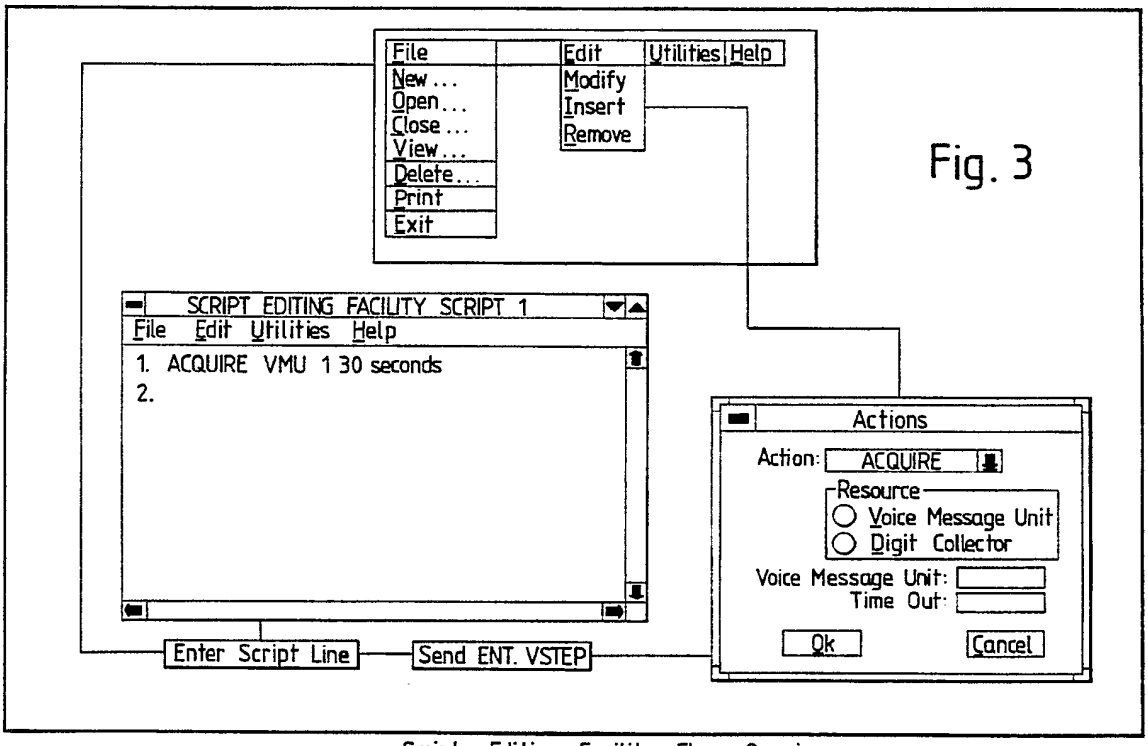


Fig. 3

Script Editing Facility Flow Overview

At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

This print takes account of replacement documents submitted after the date of filing to enable the application to comply with the formal requirements of the Patents Rules 1990.

This print incorporates corrections made under Section 117(1) of the Patents Act 1977.

GB 2 273 418 A

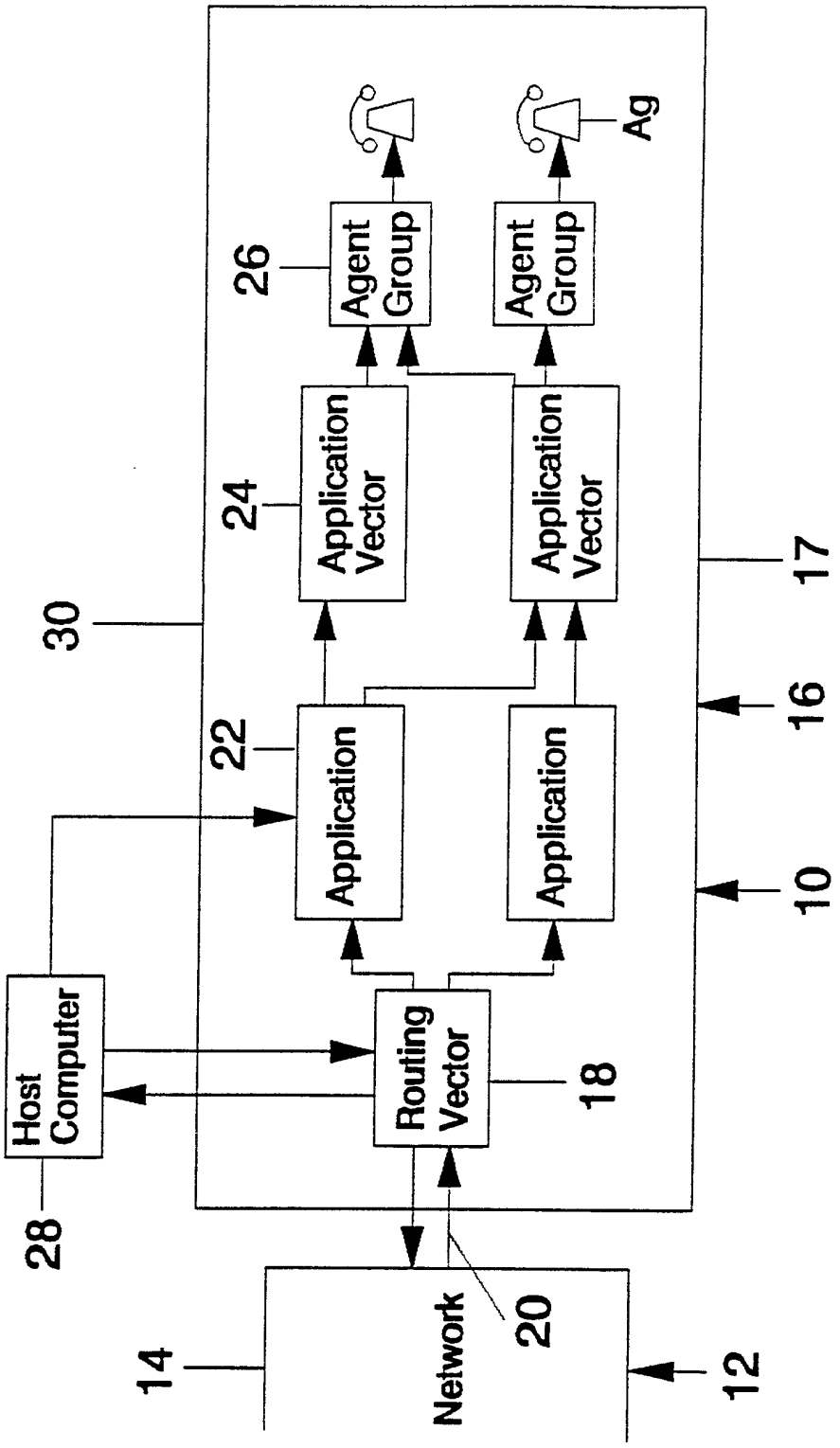

Host Computer
28
Routing Vector
18
Application
22
Application Vector
24
Agent Group
26
Ag
30
17
16
10
Network
20
14
12

Fig. 1

Fig. 2

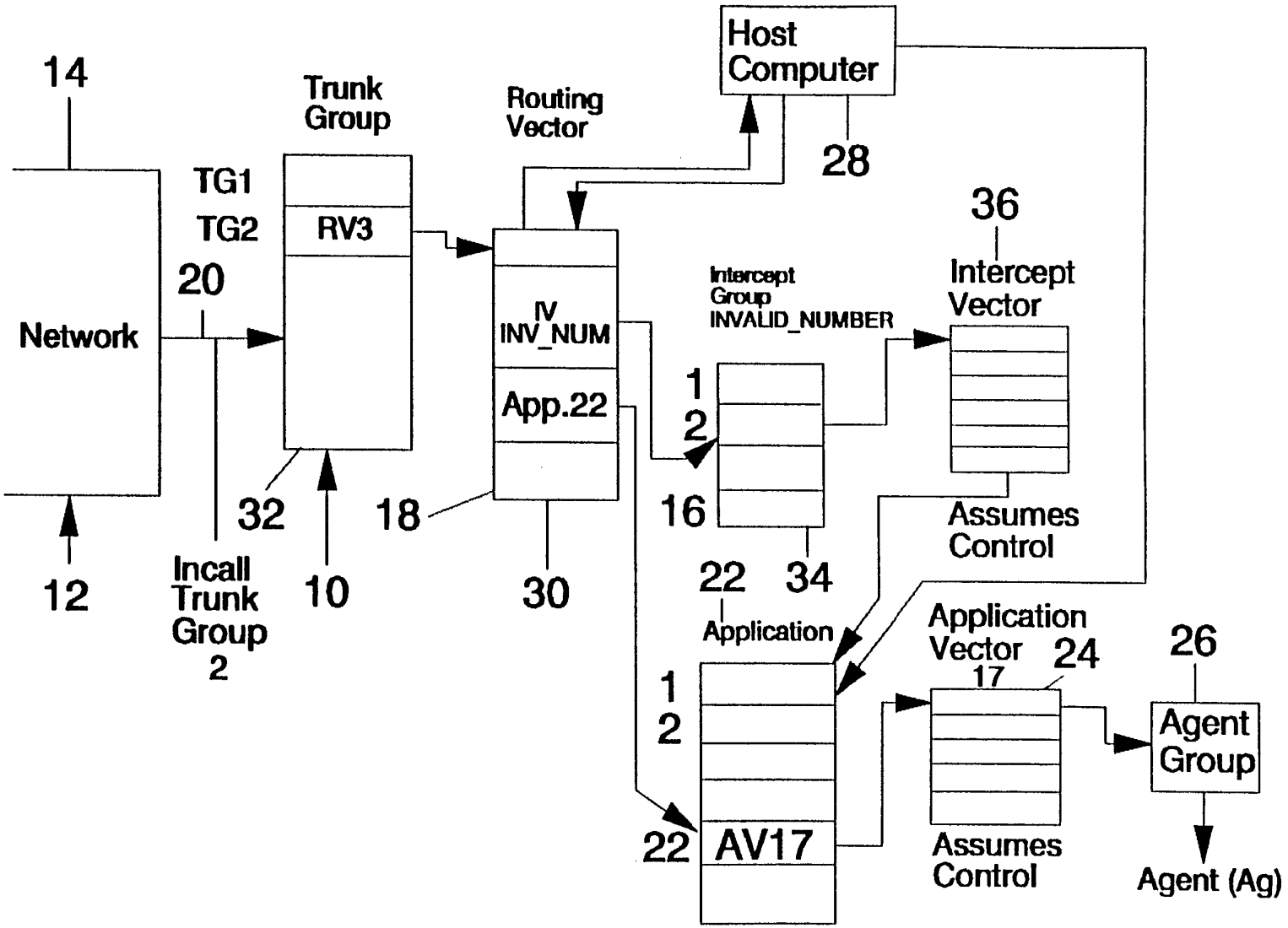
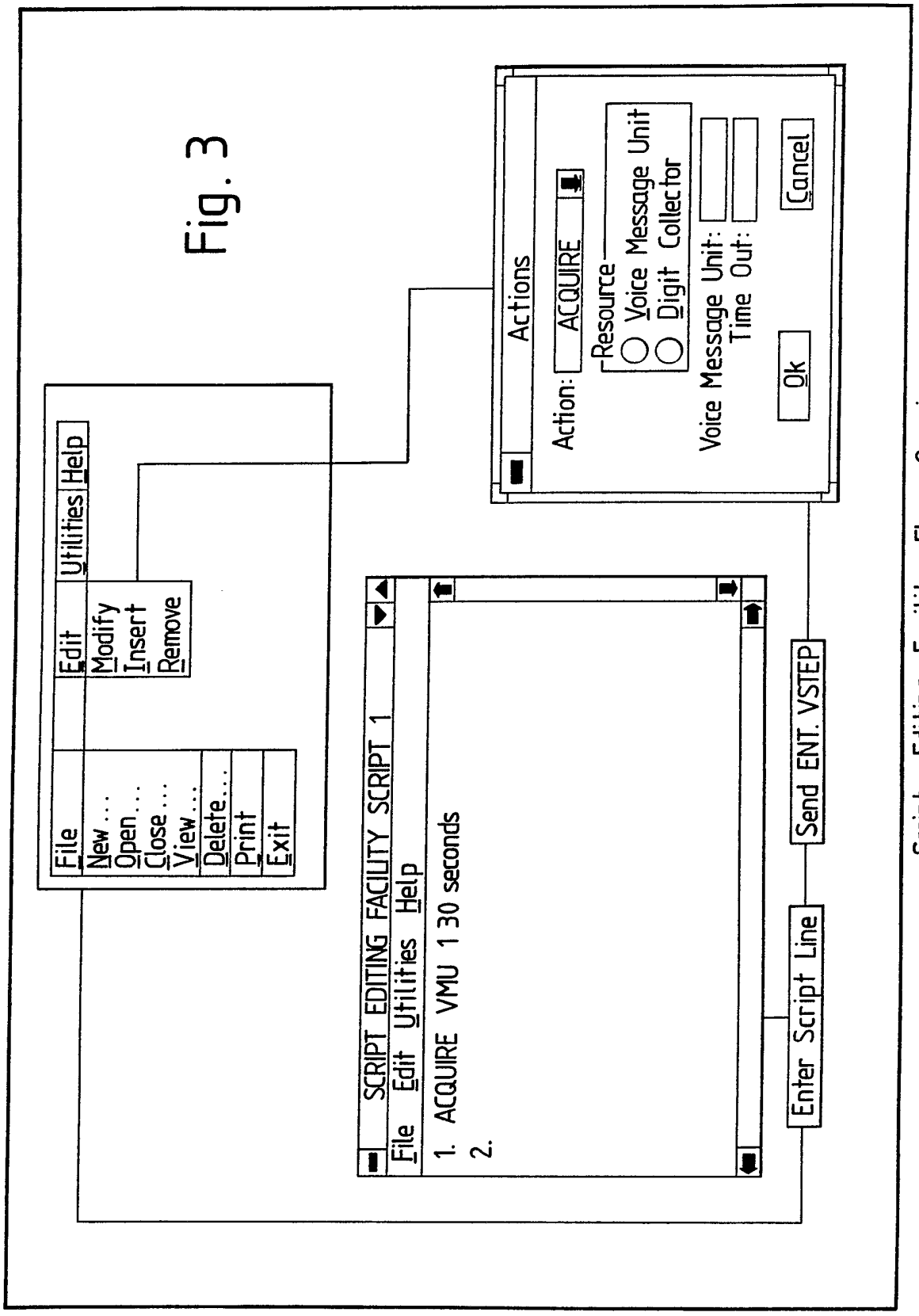
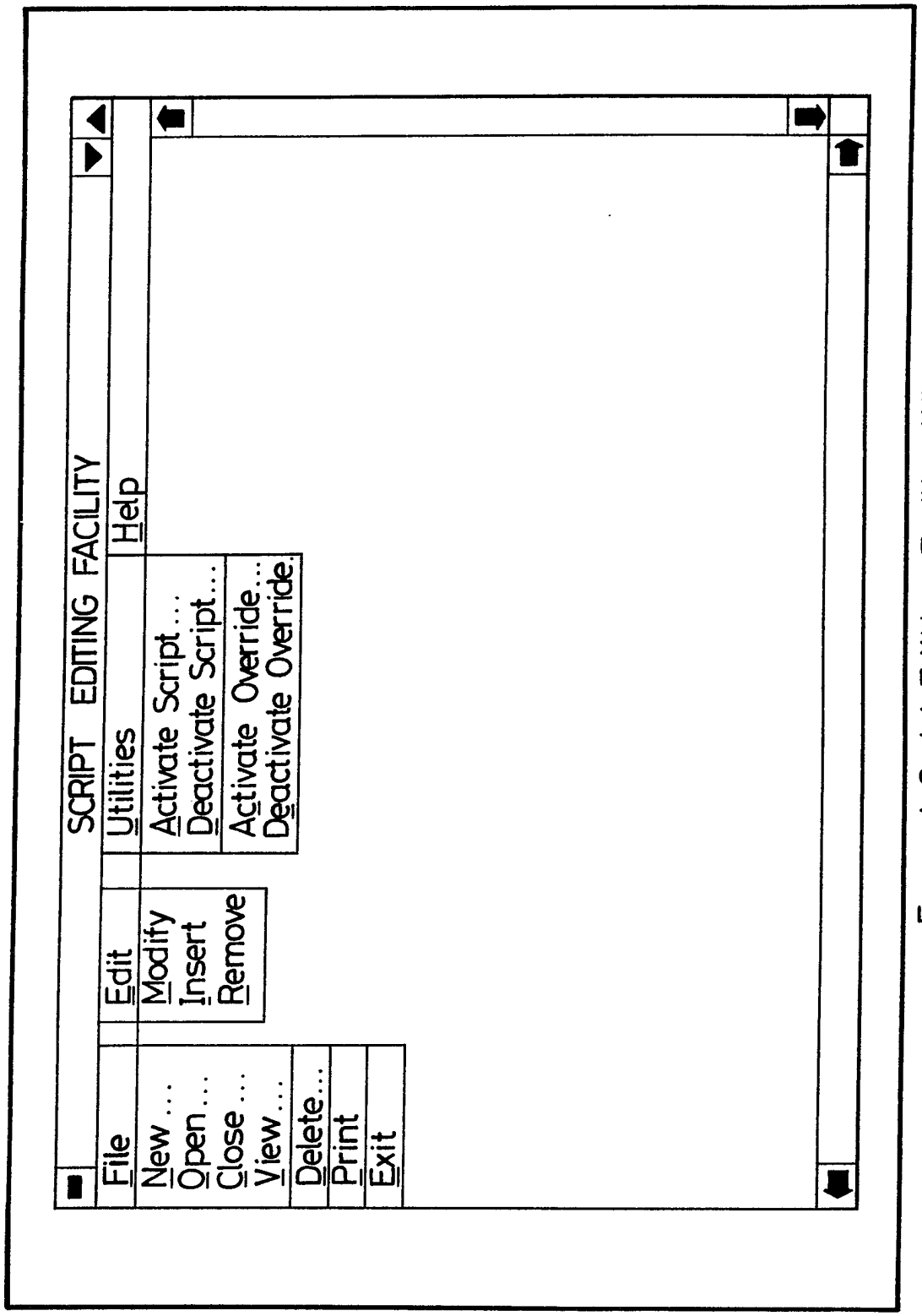

14
Trunk Group
Routing Vector
Host Computer
28
TG1
TG2
RV3
20
36
Intercept Vector
Intercept Group INVALID_NUMBER
IV INV_NUM
App.22
Network
1
2
16
Assumes Control
32
18
12
Incall Trunk Group 2
10
30
22
34
Application
Application Vector
17
24
26
1
2
22
AV17
Agent Group
Assumes Control
Agent (Ag)

Fig. 3



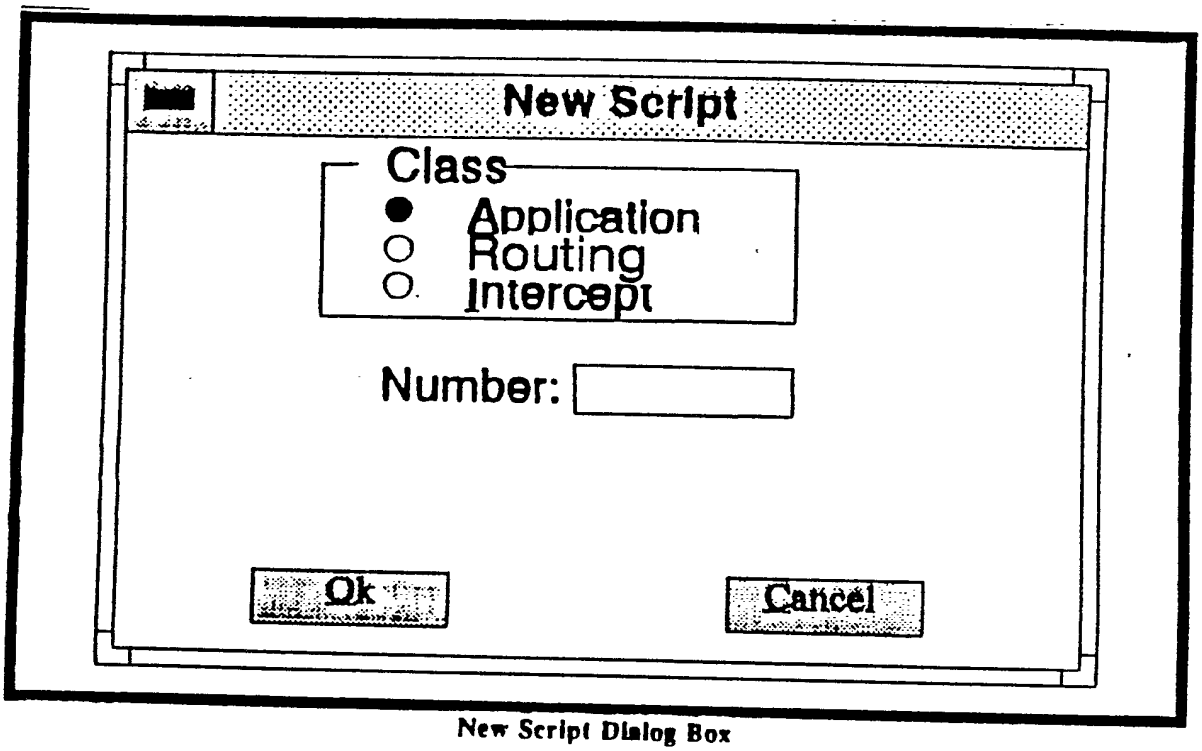
Script Editing Facility Flow Overview



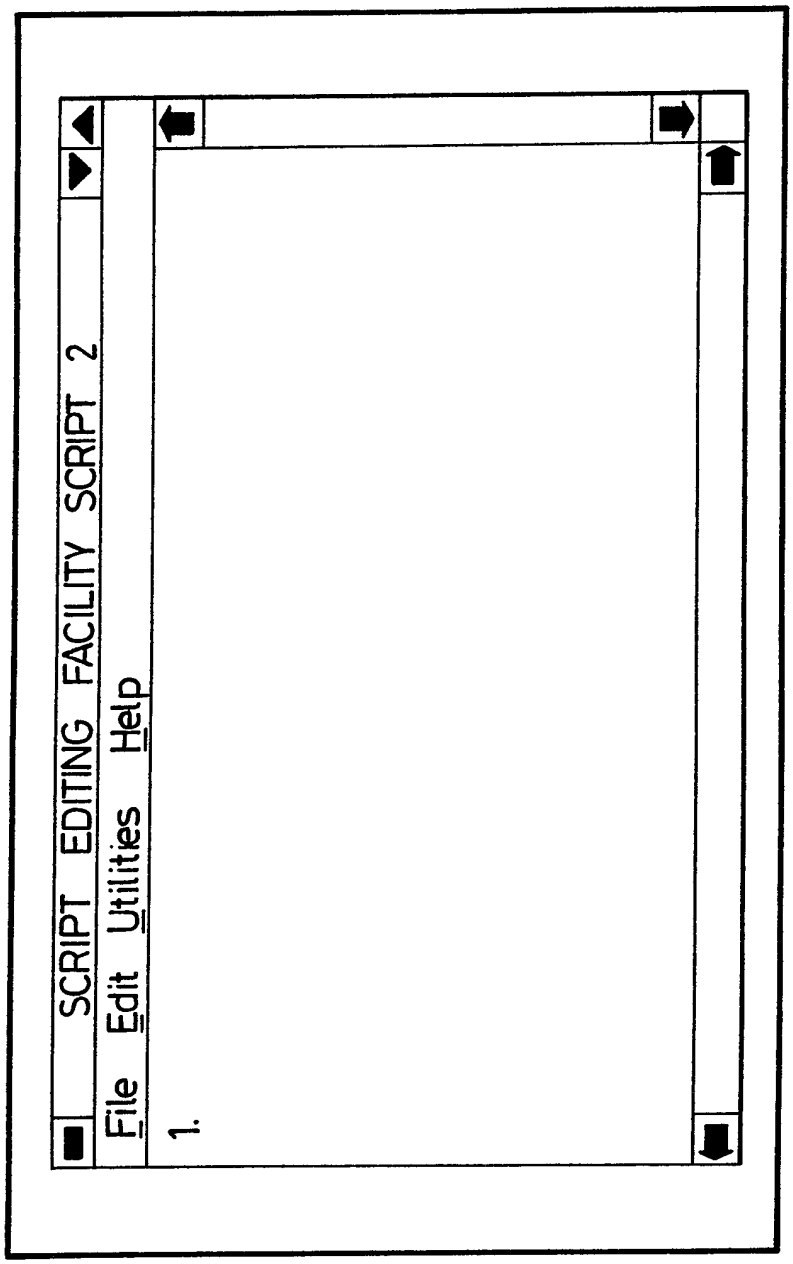
Exposed Script Editing Facility Window

Fig. 4

Fig. 5



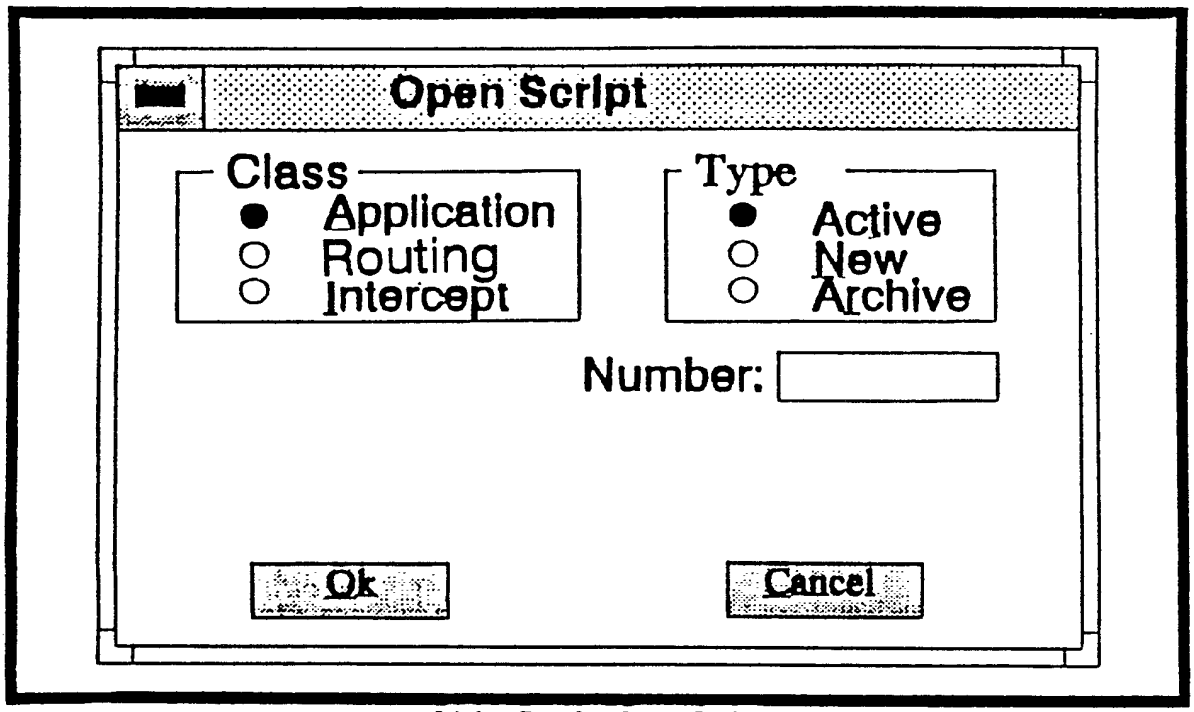
New Script Dialog Box



SEF New Script Window

Fig. 6

Fig. 7



Dialog Box for Open Script

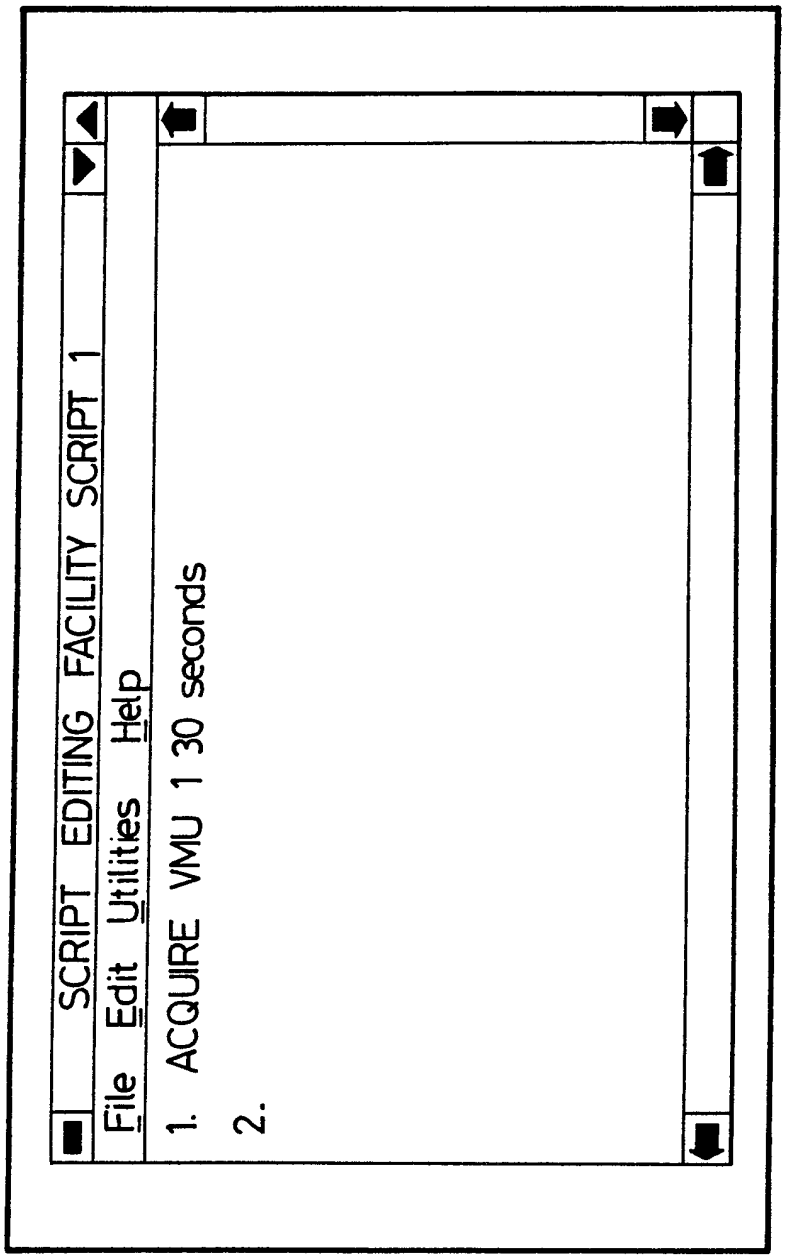

SCRIPT EDITING FACILITY SCRIPT 1
File Edit Utilities Help
1. ACQUIRE VMU 1 30 seconds
2.
Script Edit Facility Open Script Window

Fig. 8

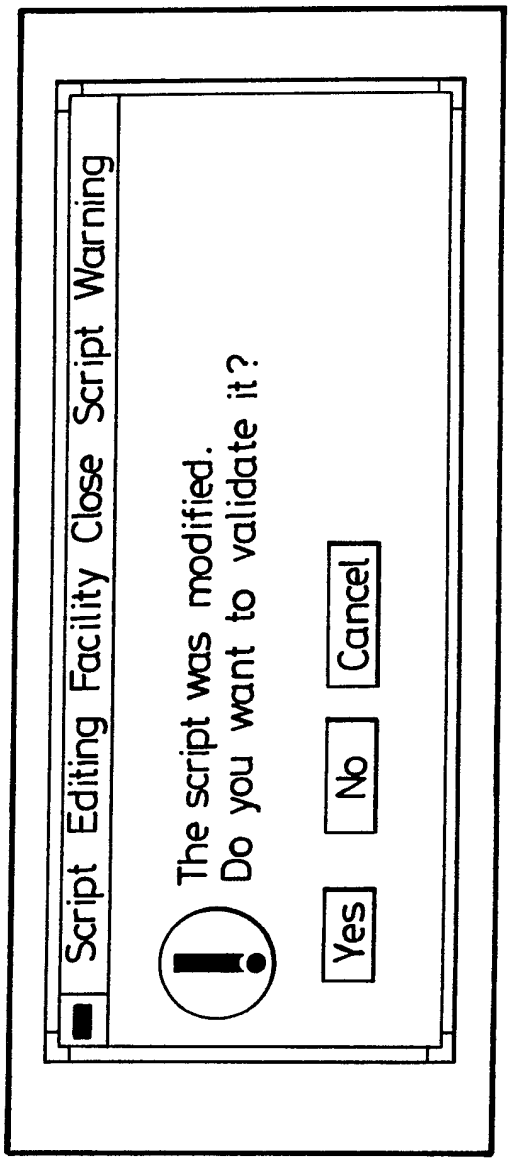
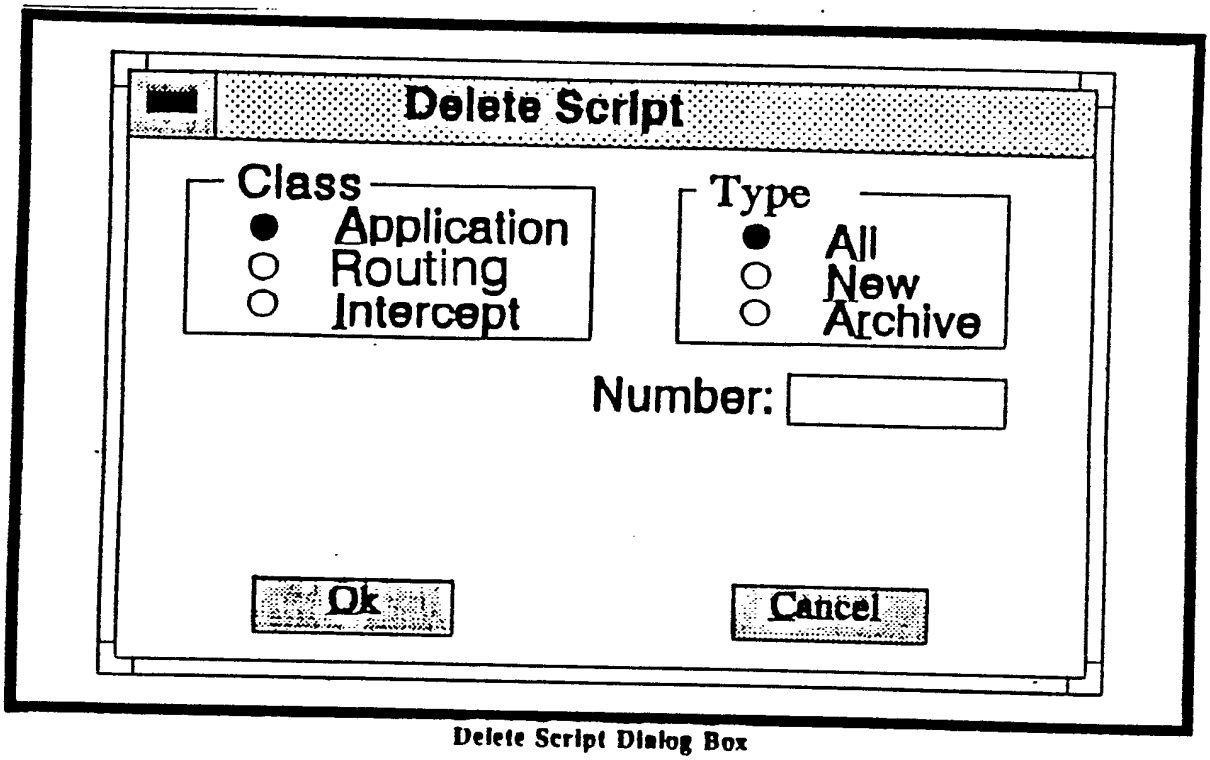

Script Editing Facility Close Script Warning
The script was modified.
Do you want to validate it?
Yes
No
Cancel
SEF Close Script (edit)

Fig. 9

Fig. 10



Delete Script Dialog Box

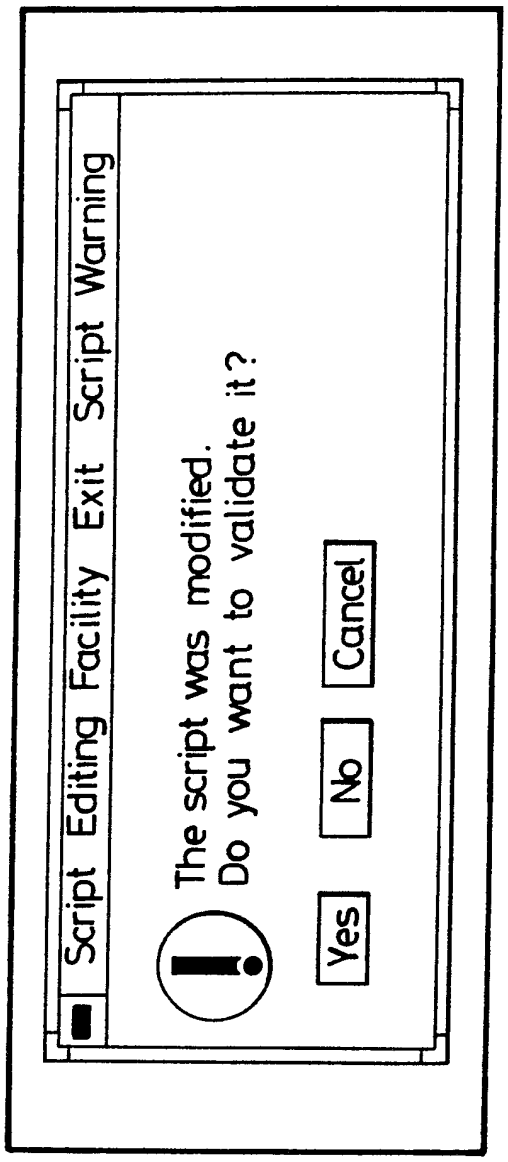
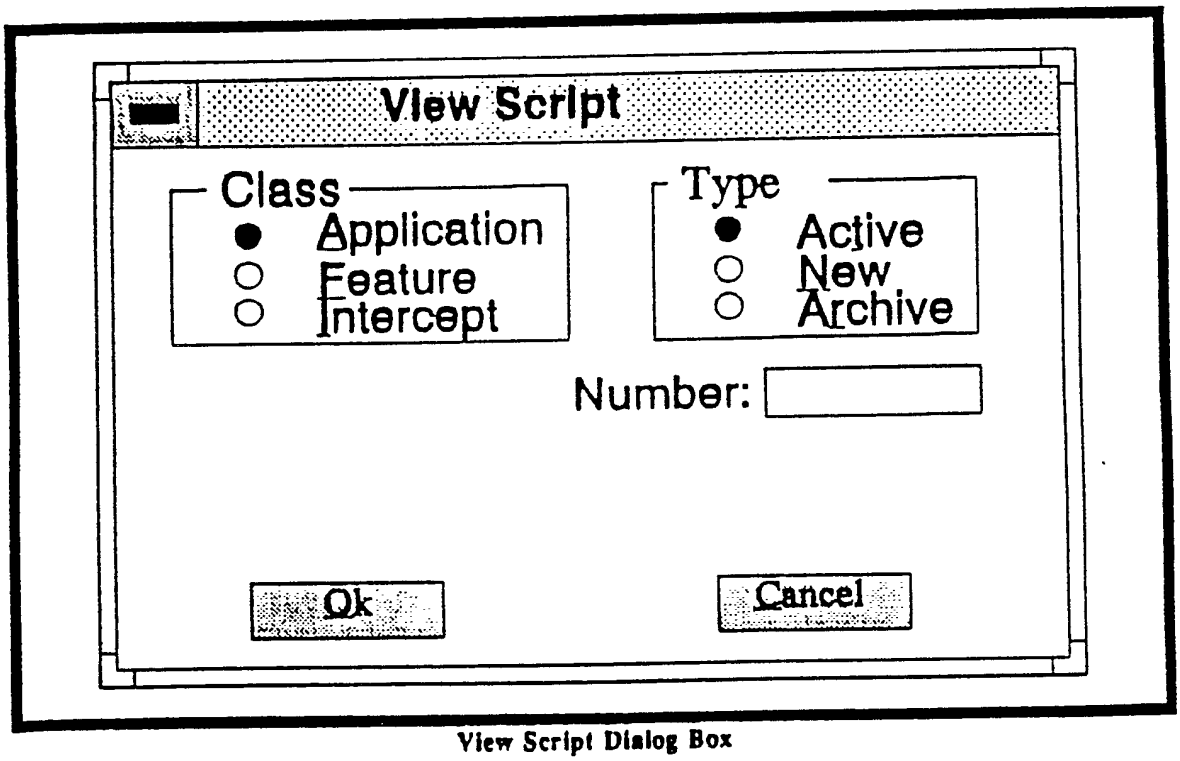
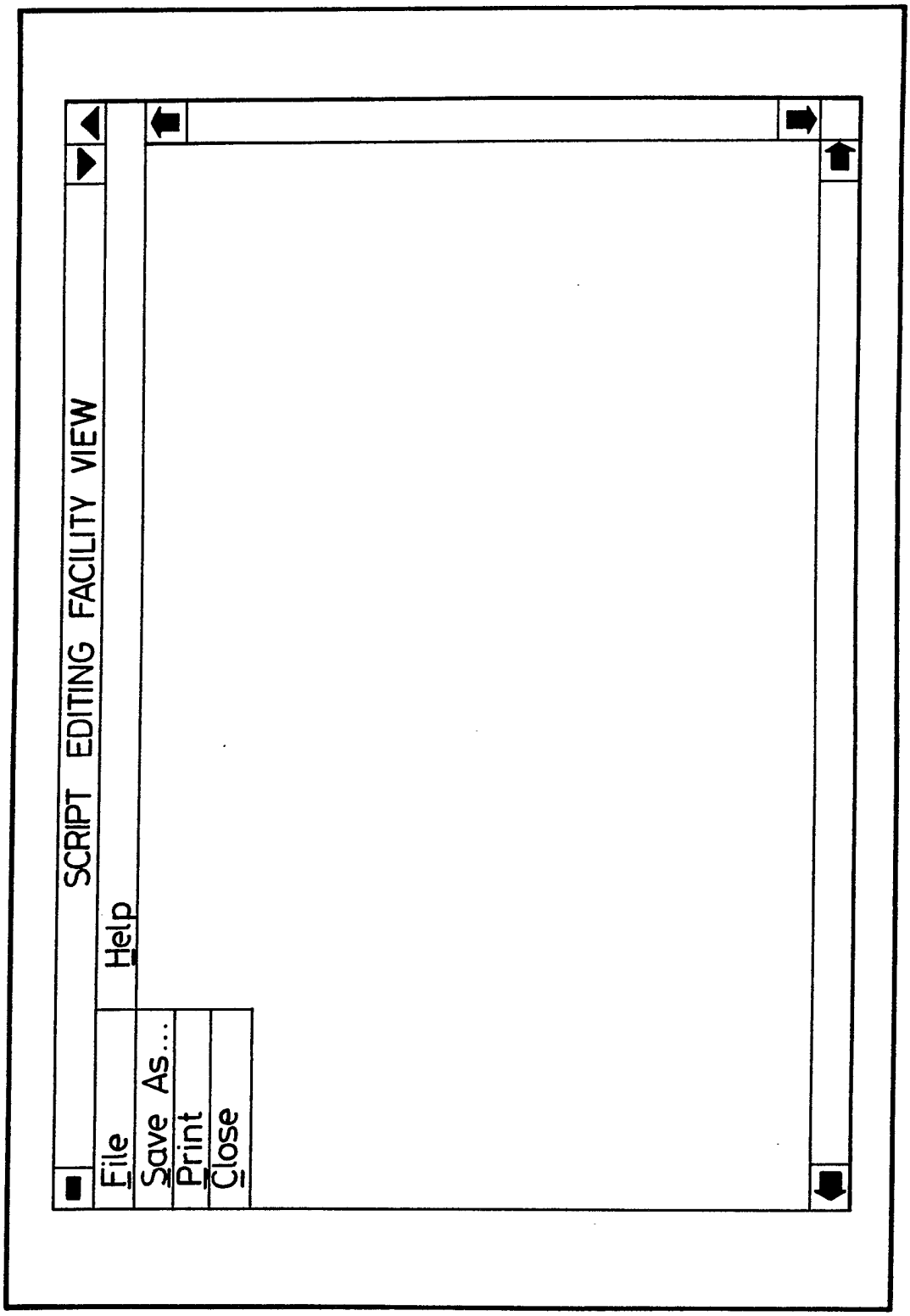

Script Editing Facility Exit Script Warning
The script was modified.
Do you want to validate it?
Yes
No
Cancel
Quit Script (an edit was made)

Fig. 11

Fig. 12



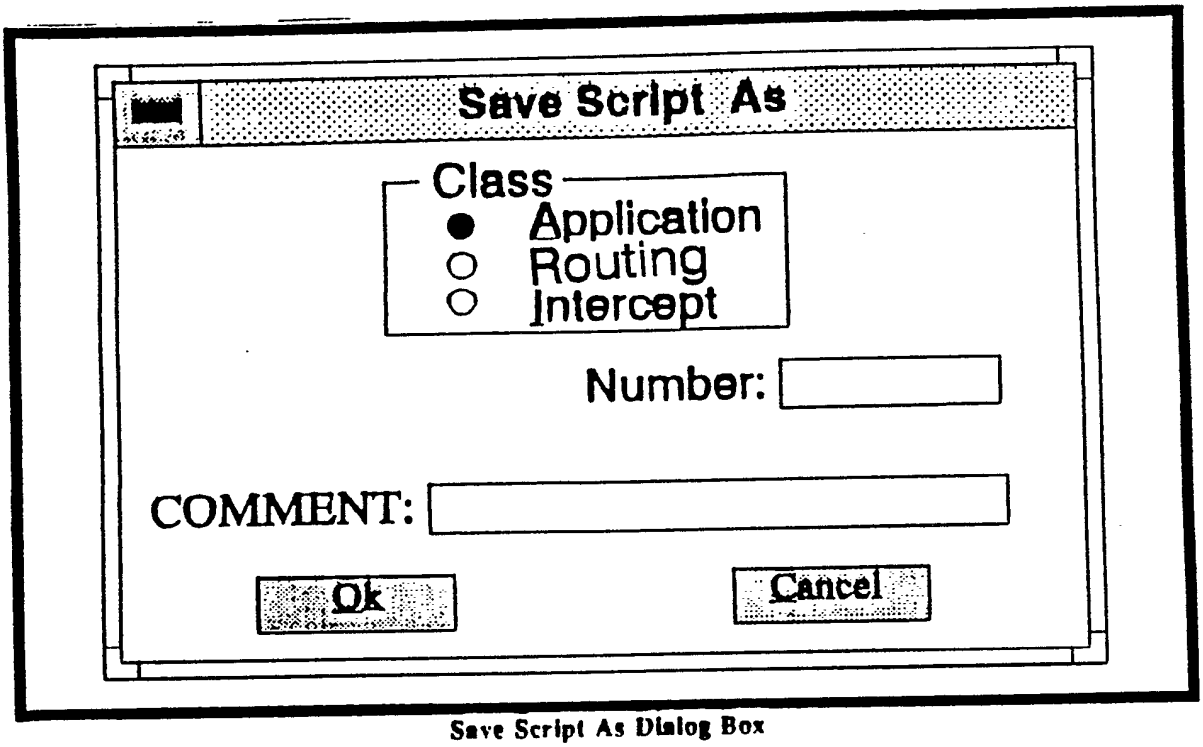
View Script Dialog Box



Script Editing Facility View Window

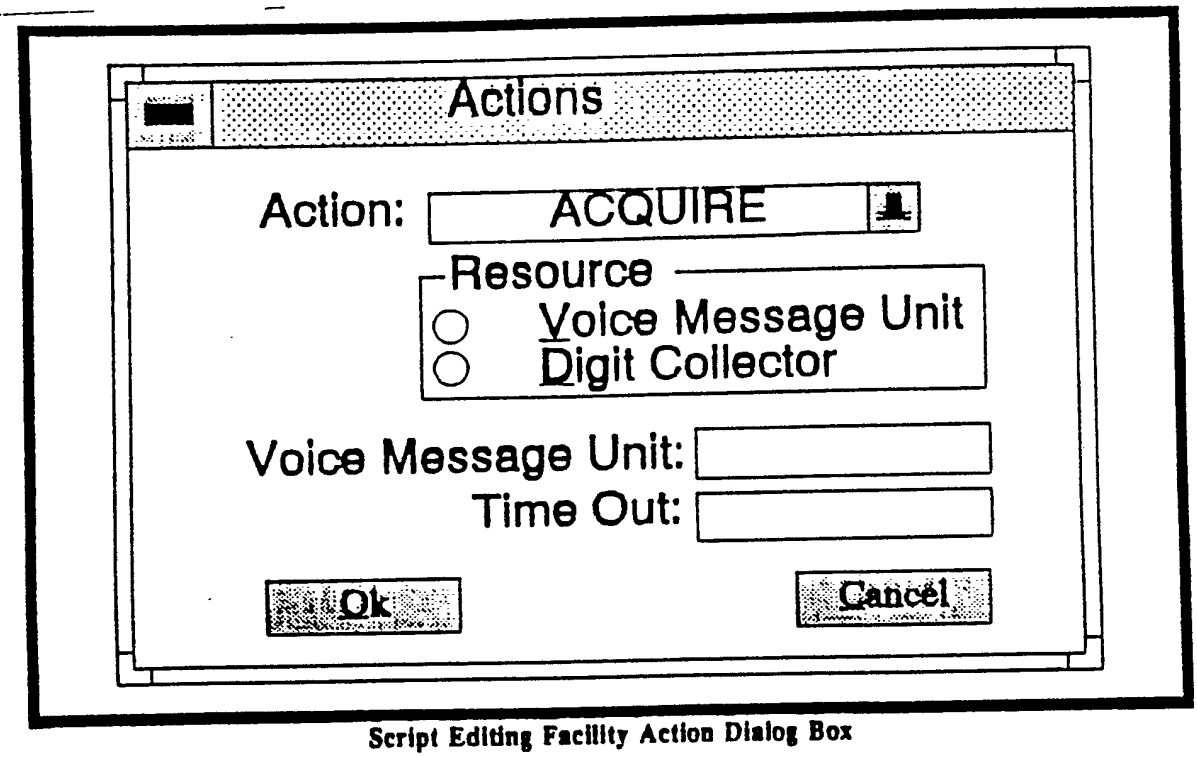
Fig. 13

Fig. 14



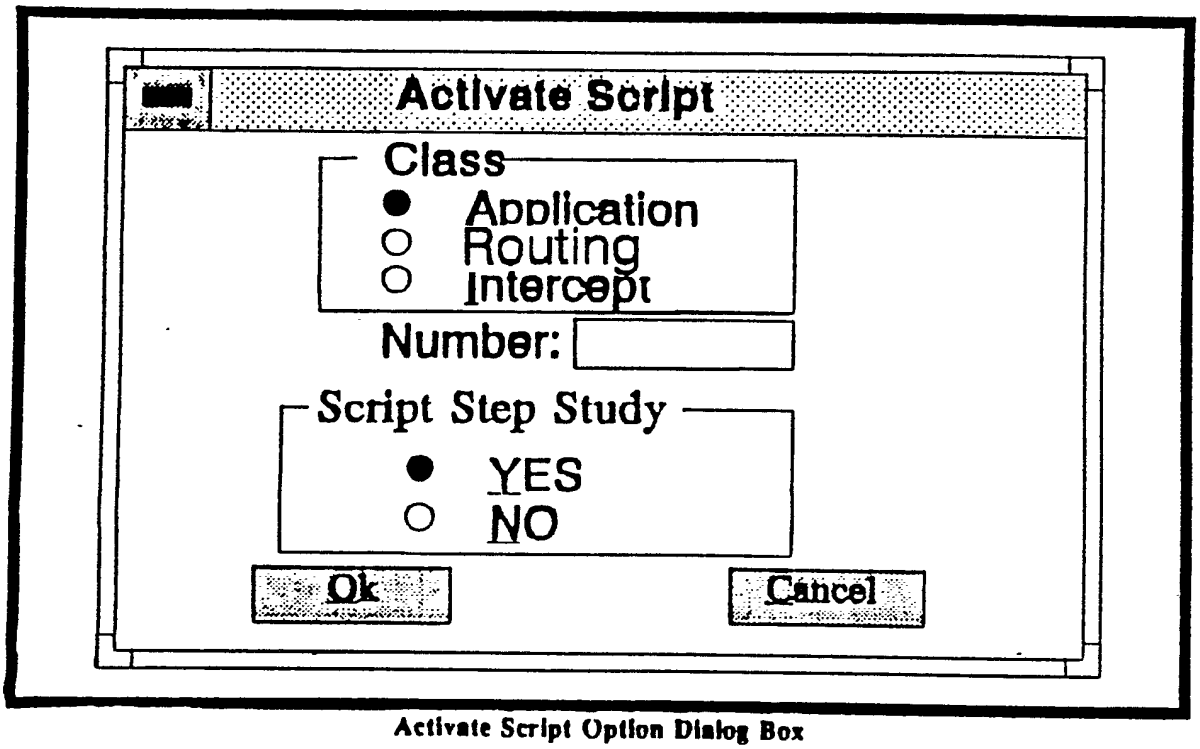
Save Script As Dialog Box

Fig. 15



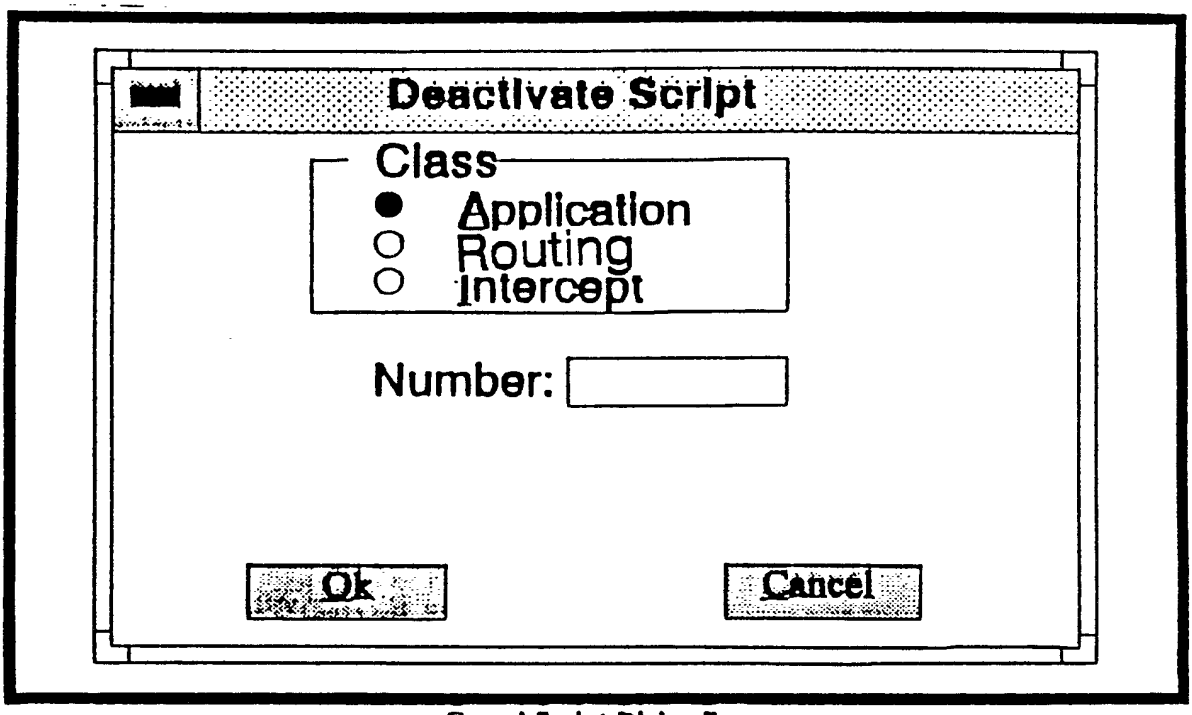
Script Editing Facility Action Dialog Box

Fig. 16



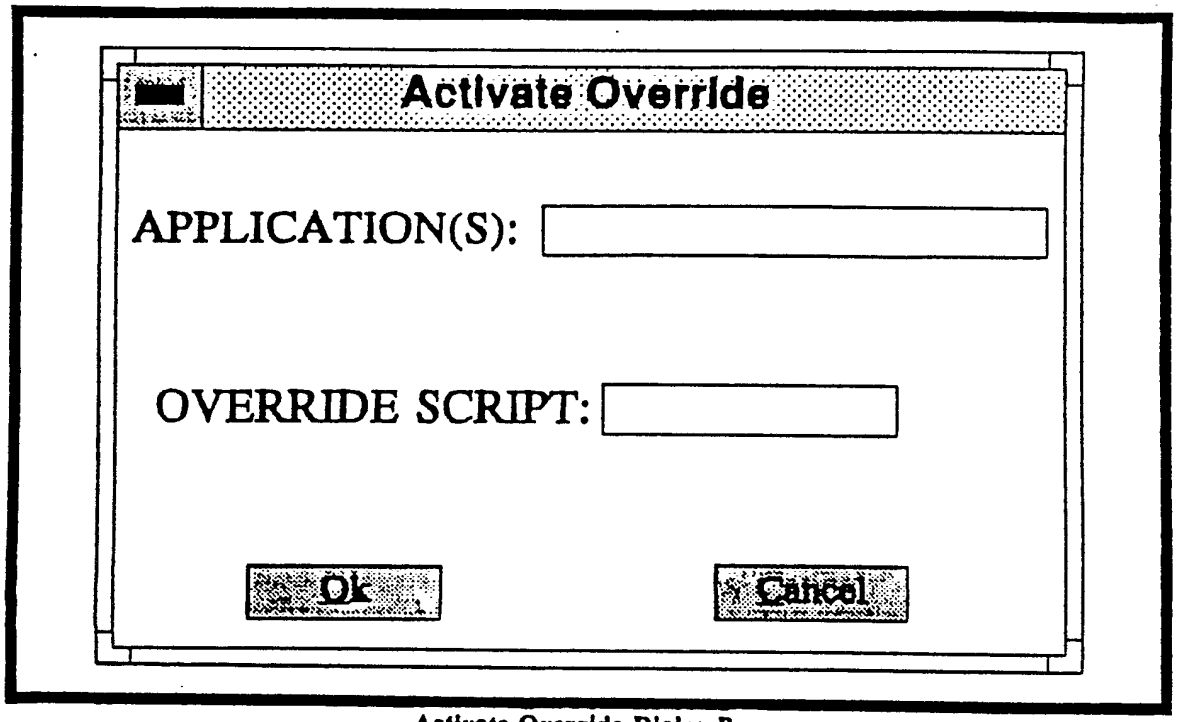
Activate Script Option Dialog Box

Fig. 17



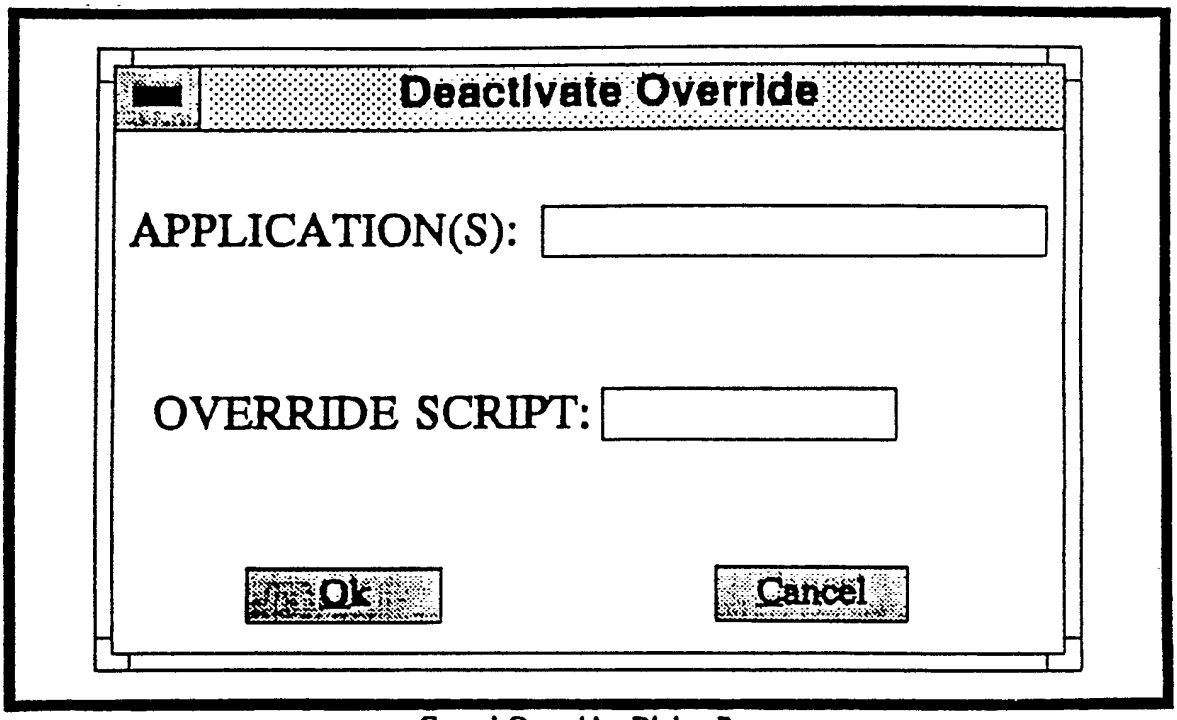
Cancel Script Dialog Box

Fig. 18



Activate Override Dialog Box

Fig. 19



Cancel Override Dialog Box

DEVICE FOR PROGRAMMING SCRIPT SETS IN A TELEPHONE SYSTEM

BACKGROUND OF THE INVENTION

The present invention relates to programming devices for a telephone system.

In the past, telephone systems have been provided for handling incoming calls by a user. However, these systems have usually been hard-wired by the manufacturer, and do not provide sufficient flexibility for the customer's need at the site of the user. Thus, it is desirable to provide such systems which may be modified in accordance with the user's needs.

SUMMARY OF THE INVENTION

A principal feature of the present invention is the provision of a programming device for a telephone system.

The device of the present invention is programed by the user with script sets, and comprises means for manipulating the script sets, means for editing the script sets, and means for performing tools for the script sets.

A feature of the invention is that the user may initialize and modify operation of the telephone system in accordance with the users needs.

Another feature of the invention is that the user may readily program the device using the script sets to form the system in accordance with the requirements of the user.

A further feature of the invention is that the user may form the desired system at the site of the user by programming the script sets.

Another feature of the invention is that the program includes multiple script classes which are programmed to route the incoming telephone calls.

Yet another feature of the invention is that the program includes multiple script types which may be programmed by the user.

Further features will become more fully apparent in the following description of the embodiments of the invention, and from the appended claims.

DESCRIPTION OF THE DRAWINGS

In the drawings:

Fig. 1 is a block diagram of a control device for the network of a telephone switching system;

Fig. 2 is another block diagram of the control device of Fig. 1;

Fig. 3 is a diagrammatic view of a script editing facility flow overview in a programming device for the telephone system of the present invention;

Fig. 4 is a diagrammatic view of an exposed script editing facility window for the device of Fig. 3;

Fig. 5 is a diagrammatic view of a new script dialog box;

Fig. 6 is a diagrammatic view of a new script window;

Fig. 7 is a diagrammatic view of a dialog box for open script;

Fig. 8 is a diagrammatic view of a script edit facility open script window;

Fig. 9 is a diagrammatic view of a close script;

Fig. 10 is a diagrammatic view of a delete script dialog box;

Fig. 11 is a diagrammatic view of a quit script;

Fig. 12 is a diagrammatic view of a view dialog box;

Fig. 13 is a diagrammatic view of a script editing facility view window;

Fig. 14 is a diagrammatic view of a save script as dialog box;

Fig. 15 is a diagrammatic view of a script editing facility action dialog box;

Fig. 16 is a diagrammatic view of an activate script option dialog box;

Fig. 17 is a diagrammatic view of a cancel script dialog box;

Fig. 18 is a diagrammatic view of an active override dialog box; and

Fig. 19 is a diagrammatic view of a cancel override dialog box.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Referring now to Fig. 1, there is shown a control device generally designated 10 for a telephone switching system generally designated 12. The telephone system 12 has a network 14 which is connected to a switching system 16, such as an Automatic Call Distributor (ACD) 17, included in the control device 10.

The control device 10 has a routing vector or script 18 to receive incoming calls from a trunk 20 connected to the network 14 of the telephone system. The routing vector 18 may route calls to an application 22 which in turn directs the calls to an application vector 24. For purposes of this application, a vector is considered to be one or more statements or instructions in the program for a computer or similar device. The application vector 24 routes the calls from the application 22 to one or more agent groups 26 for handling by one or more agents Ag associated with the agent groups 26.

As shown, the routing vector 18 is also connected to a host computer 28 which may process calls coming into the ACD. The host computer 28 may direct the incoming calls to the application 22, as will be seen below.

The ACD of the control device 10 may comprise a

suitable computer 30 or Central Processing Unit (CPU) having a random access memory (RAM) and Read Only Memory (ROM) for processing information related to the incoming telephone calls, and route the incoming telephone calls. As will be seen below, the routing vector 18, as well as the application 22 and application vector 24, may be controlled by the user at the location of the control device 10, by programming the computer 30 with high level statements in a high level language for the computer 30 to make control of the control device 10 relatively simple to define.

With reference to Fig. 2, an incoming call from the network 14 is presented to the control device 10 on a trunk group 2 of a trunk group 32 having a plurality of separate locations for processing the calls as defined by the user through suitable programming of the device 10. As shown, the trunk group 2 is programmed to direct the calls to Routing Vector 3 of the routing vector 18 having a plurality of possible locations. In this case, the call is presented to routing vector 3, as defined in the trunk group 32.

The routing vector 3 then executes a series of defined steps which is accomplished by previous programming of the computer 30. Such steps may include a jump to an intercept group 34 in the case of a fault

condition, with the intercept group 34 having a plurality of separate locations as defined by programming the computer 30. The intercept group 34 directs calls to an associated intercept vector 36 having a plurality of separate locations, with the intercept vector 36 being based upon the group and intercept class of the call. The intercept vector 36 then assumes control of the call. In the specific example shown, the routing vector 3 jumps to intercept group INVALID NUMBER. The INVALID NUMBER group and the calls assigned intercept class translate to intercept vector 1.

In another case, the routing vector 18 may direct the calls to the application 22 having a plurality of separate locations which are defined by the user through previous programming of the computer 30. As shown, the routing vector 18 transfers the calls to application 22 which in turn routes the call to an application vector 24 having a plurality of locations, after which the application vector 24 assumes control of the calls. In turn, the application vector 24 may direct the call to one or more of the agent groups 26 for handling of the call by one or more agents Ag associated with the agent groups 26. If desired, one or more locations of the intercept vector 36 may transfer the calls to one or more locations of the application 22 which ultimately directs the call to the application vector 24 and the agent groups 26.

In a preferred form, each trunk 20 is assigned to an intercept class, and may point to the same or different routing vectors 18. The intercept vector 36 utilized in the case of a failure may usually constitute an announcement or recorded tone, as defined by the user. Also, the application vector 24 may include an announcement, and ultimately directs the calls to the agent group 26.

As shown in Figs. 1 and 2, the routing vector 18 is connected to the host computer 28, and informs the host computer 28 of an incoming call. The routing vector 18 may request information concerning the call from the host computer 28, and the host computer 28 may request information concerning the call from the routing vector 18 which is obtained by the routing vector 18 from the network 14, which information is supplied by the routing vector 18 to the host computer 28. Also, the application vector 18 may request the host computer 28 to route a given call depending upon the nature of the call, as defined by the user, and the host computer 28 may direct the calls to the application 22 for ultimate disposition at the agent group 26.

The routing of calls is defined by information supplied by the network 14 which may be a Trunk Group

which is the number of the inbound trunk group on which the call was presented, an Internal Directory Number (DID) to which the call should be presented, a Dialed Number Identification Service (DNIS) which identifies the called number, or an Automatic Number Identification (ANI) which comprises an area code, an area code + exchange code, or an area code + exchange code + station address. Thus, the control device 10 routes the calls using this information associated with an incoming call for routing by the ACD or host computer 28.

The ACD provides an interaction with the network 14 of the telephone system 12 for improved handling of the incoming calls. When a valid call has been received by the routing vector 18, the routing vector 18 retrieves information concerning the call from the network 14 which is collected by the trunk 20. The routing vector 18 is responsible for any error handling during collection of digits from the network 14. Each trunk group 32 has a routing vector 18 and route class assigned to it.

The routing vector 18, the intercept group 34, the intercept vector 36, the application 22, and the application vector 22 essentially serve as different switches to direct calls in the ACD. Thus, in accordance with the present invention, the user may define the interaction of these switches alone, or in combination

with the host computer 28 in a simplified manner through use of the high level statements in programming the computer 30, as will be seen in the Examples below.

EXAMPLE I

The following is a simple example of a call being routed, as programmed by the following statements for the computer 30, of a call being routed based upon the ANI collected from the network 14. No check is being made here on whether the network information is collected successfully. In this case, if the network information retrieval fails or the ANI translation fails, the ROUTE TRANSLATION will make use of the default routing information established at the point the vector was accessed, i.e., the trunk group's routing information.

(1) RETRIEVE NETWORK

(2) INFORM HOST ON ARRIVAL

(3) TRANSLATE ANI

(4) ROUTE TRANSLATION

EXAMPLE II

The following is an example of a route on DNIS with the "ANI Required Indicator" being checked. The default routing information was established prior to the activation of the routing vector 18 and is that of the trunk group 32.

Upon entry, the trunk 20 is requested to RETRIEVE the network information. If a failure occurs while retrieving the network information, control is passed to the intercept vector 36 associated with the intercept group Invalid Procedure and the call's intercept class.

On a successful network data collection, the vector INFORMs the host computer 28 of the call's arrival and TRANSLATES the DNIS to its routing information. If the translation fails, the vector script ROUTEs the call based upon the initial translation of the trunk group information.

On a successful DNIS translation, the vector script checks if ANI is required for the DNIS number. If it is required, the script REQUESTs the ANI from the network 14. If the ANI request fails, the call is sent to the intercept vector 36 specified by the call intercept class and the intercept group 34 Invalid Number.

Upon either ANI not required or successful ANI retrieval, the call is routed based upon the DNIS route translation.

```
(1) RETRIEVE NETWORK INFORMATION
(2) IF RESULT EQ SUCCESS GO TO 4
(3) INTERCEPT INVALID_PROCEDURE
(4) INFORM HOST ON ARRIVAL
(5) TRANSLATE DNIS
(6) IF RESULT EQ SUCCESS GO TO 8
(7) ROUTE TRANSLATION
(8) IF ANI_IND EQ ANI_NOT_REQUIRED GOTO 12
(9) REQUEST NETWORK ANI_IND
(10) IF RESULT NE FAIL GOTO 12
(11) INTERCEPT INVALID_NUMBER
(12) ROUTE TRANSLATION
```

EXAMPLE III

In the following example, if the vector script 18 is unable to retrieve the network information and translate the ANI into routing information, the call is treated with intercept handling. If the call's ANI is collected and translated successfully, the vector script checks if the ANI is flagged as being a candidate for host (host computer 28) routing. If not, the script informs the host of the arrival and routes the call based upon the ANI translation.

If the ANI is flagged as preferring host routing, the host is provided with a call arrival indication and REQUEST for instruction. The vector waits 2 seconds for a host route.

If the host fails to respond or the message is unable to be transmitted to the host, the call is routed based upon the previous ANI translation.

If the host responds within the 2 seconds, the vector "translates" the host provided routing information and the subsequent ROUTE TRANSLATION makes use of the host provided routing information to route the call.

```
(1) RETRIEVE NETWORK INFORMATION
(2) IF RESULT EQ SUCCESS GOTO 4
(3) INTERCEPT INVALID_NUMBER
(4) TRANSLATE ANI
(5) IF RESULT EQ SUCCESS GOTO 7
(6) INTERCEPT INVALID_NUMBER
(7) IF HOST_ROUTE_PREFERRED EQ YES GOTO 10
(8) INFORM HOST ON ARRIVAL
(9) GO TO 11
(10) REQUEST HOST INSTRUCTION ON ARRIVAL 2 SECONDS
(11) ROUTE TRANSLATION
```

Thus, a program of simplified form may be programmed by the user through use of the statements by the user at the location of the control device 10.

With reference to Fig. 3, the activation of a script editing session within a Script Editing Facility (SEF) window will prevent the user from accessing a Terminal Emulation window, even when the SEF window with an open script is minimized (iconized). The program for the vectors or script provides the capability to define a single SEF window. Access to the SEF window is allowed through the Command window's SEF icon. Within the SEF window, the user has the ability to define and modify a set of scripts, by the use of the dialogued boxes. Script sets are stored in the system, making it possible for the

script sets to be easily accessed by other terminal users. Also, the user is able to launch and cancel scripts from the SEF window.

All of the usual Presentation Manager window manipulation controls are available to the user to a low effective management of the monitor display space. The user is able to minimize the window while attention is being focused on other windows. Minimizing the window will reduce to an icon on the screen. Closing the window cancels the script editing session, minimize the SEF window, and return focus to the Main Command window.

The SEF user interface is a line editor driven by menu bar selections and dialog box interactions. A window is presented to the user for the monitoring and the selection (via pointing and clicking or cursoring) of script step edits. Fig. 3 demonstrates the flow of "creating a new vector or script." Prior to this flow the user had selected the Script menu option "New." This procedure causes retrieve script and open script commands to be sent to the Automatic Call Distributor (ACD). After the user's "Script 1" window is opened, the user may select, via choosing SEF's menu Edit selection Insert option, a script Action type dialog box. In this example, the user has selected and "OK'd" the "ACQUIRE VMU" action command. When "OK'd", the "Script Editing Facility Script

1" window is updated. After this is accomplished, the user may choose to continue the editing session or quit the Script menu option "Exit."

With reference to Fig. 4, this window depicts the main SEF menu bar with all drop down selections exposed. At this level the user may perform a SEF editing session. This procedure includes manipulating script sets (File), script steps (Edit), and script tools (Utilities). The menu selections in this window operate as follows:

File: This selection enables the user to work with Scripts. The user of this selection's options has the ability to:

1. create scripts (New)
2. read an existing script and display it for modification (Open)
3. remove from the screen the active window and all associated dialog boxes (Close)
4. review another existing script (View)
5. remove an existing script (Delete)
6. prepare and schedule a script for printing (Print)
7. end a session (Exit)

Edit: With this selection the user is able to change

(Modify) original script steps, create (Insert) new script steps, and delete (Remove) original script steps.

Utilities: Using the options in this selection, the user may "turn on" (Activate Script) or "turn off" (Deactivate Script) a script. Also, the user may make a replacement of a script associated with an application (Activate Override), or rescind the association of a script with an application (Deactivate Override).

The manipulation of script sets is accomplished using the File selections: New, Open, Close, View, Delete, Print, and Exit.

The selection of the File option New allows the user to enter a new script via the dialog box shown in Fig. 5. This dialog box allows the user to create a script description for the new script that is to be created. The user accomplishes this activity by:

1. selecting via the Class group box selection field, the required one of three classes for the script
2. entering the identifier of the new script in the Number: entry field

The pushbutton controls in this dialog box operates as follows:

1. OK: If the number does not exist in the system, this pushbutton will close the dialog box and initialize a New Script Window.
   If the number does exist, this pushbutton causes a warning message that will inform the user of the undesirable situation.
2. Cancel: This control closes this dialog box with no other action.

After filling in the entry field and pushing the "OK" button, the user is presented with the window shown in Fig. 6. The user will see the output of newly inserted script steps in this window.

The selection of the File option Open will produce the dialog box shown in Fig. 7. The user fills out this dialog box to open an existing Script. This dialog box allows the user to create a script description for an existing script that is to be edited. The user accomplishes this activity by:

1. selecting via the Class group box selection field, the required one of three classes for the script

2. selecting via the Type group box selection field, the required one of three script types for the script
3. entering the identifier of the new script in the Number: entry field

The pushbutton controls in this dialog box operates as follows:

OK: If the Number does exist in the system, this pushbutton closes the dialog box and initializes an Open Script Window with the Script displayed with step numbers and a blank line as the last step.
If the Number does not exist, this pushbutton causes a warning message that informs the user of the undesirable situation.

Cancel: This control closes this dialog box with no other action.

After filling in the entry field and pushing the "OK" button, the user is presented with the window shown in Fig. 8. After this window is displayed, the user may use either a mouse or cursor keys to select the line which may be edited. By default, the initial cursor position is on a new blank line at the end of the script allowing the user

to append new lines to the script.

Selecting File option Close will check to see if any changes had been made to the current script. If no changes had been made, the system will issue a Close command and clear the client area of the window. If changes had been made, the system displays the message box shown in Fig. 9. This warning box offers the user three options. If the user selects Yes, the system sends a command to the ACD to validate the script. If the user selects No, the system issues a close command to the ACD without validating the script. If the user selects Cancel, the system returns the user to the Editing session without closing and validating the script.

In the current SEF window, choosing the File option Delete allows the user to remove scripts from the system. The dialog box shown in Fig. 10 is presented in order to accomplish this purpose. This dialog box allows the user to create a script description for an existing script that is to be deleted. The user accomplishes this activity by:

1. selecting via the Class group box selection field, the required one of three script classes for the script
2. selecting via the Types group box selection field, the required one of three script types for

the script

3. entering the identifier of the new script in the Number: entry field

The pushbutton controls in this dialog box operates as follows:

OK: If the Number does exist in the system, this pushbutton deletes the Script from the system. If the Number does not exist, this pushbutton causes a warning message that informs the user of the undesirable situation.

Cancel: This control closes this dialog box with no other action.

The File Print option allows the user to print the script text that is currently being edited.

Choosing the File option Exit causes the system to check if any changes had been made to the current script. If no changes had been made, the system issues a Close command to the system, and clears the client area of the window. If changes had been made, the system displays the warning box shown in Fig. 11. This warning box offers the user three options. If the user selects Yes, the system sends a command to then ACD to validate the script. If the user selects No, the system issues a Close command to the

ACD without validating the script. If the user selects Cancel, the system returns the user to the Editing session without closing and validating the script.

Selecting the File View option allows the user to display, copy (Save As), and Print a different script in a secondary window. The dialog box shown in Fig. 12 is used to initiate the view window. This dialog box allows the user to create a script description for an existing script that is to be displayed. The user accomplishes this activity by:

1. selecting via the Class group box selection field, the required one of three script classes for the script
2. selecting via the Type group box selection field, the required one of three script types for the script
3. entering the identifier of the script in the Number: entry field

The pushbutton controls in this dialog box operates as follows:

OK: If the Number does exist in the system, this pushbutton closes the dialog box and displays the Script from the system in a secondary window.

If the Number does not exist, this pushbutton causes a warning message that informs the user of the undesirable situation.

Cancel: This control closes this dialog box with no other action.

After filling in the entry fields and pushing the "OK" button, the user is presented with the window shown in Fig. 13 with the selected script shown in the window so that the user may view it. From the View bar File selection, the user may choose three different options when working with the script displayed in the secondary window:

1. Save As will prompt the user via the dialog box for the identifier of the new script that is to be created. The new script is created from the current script that is displayed in the secondary View window.
2. Print prepares and schedules for printing, the current script that is displayed in the secondary view window.
3. Close removes from the screen, the current View secondary window and return control to the SEF main window.

While in the View Window, the user may copy the

displayed script to another script by selecting the Save As option and filling in the entry fields in the dialog box shown in Fig. 14. This dialog box allows the user to create a script description for a new script that is to be created from the current script which is displayed in the secondary View window. The user accomplishes this activity by:

1. selecting via the Type group box selection field, the required one of three script types for the script
2. entering the identifier of the new script in the New Number: entry field
3. optionally entering information in the Comment: entry field (displayed only for the script Class of feature)

This new script Class is predetermined by the script class of the current script that is displayed in the secondary View window.

The pushbutton controls in this dialog box operates follows:

OK: If the Number does not exist in the system, this pushbutton closes the dialog box and creates the new Script on the system.

If the Number does exist, this pushbutton causes a warning message that informs the user of the undesirable situation.

Cancel: This control closes this dialog box with no other action.

View's Print option allows the user to print the script text that is currently being viewed. Choosing View's Close option allows the user to close the SEF's View secondary window.

The next menu item in the Script Editing Facility window menu bar is the Edit option. Under this pull-down option, the user may chose to Modify, Insert, and Remove script sets. To use these options, the user must first create a new or open an existing script that the user wants to edit. Second, the user, using a mouse or the keyboard arrow keys, selects the script step in SEF's Main Window that the user wants to modify, insert, or remove. Finally, the user may select the activity (Modify, Insert, or Remove) the user wants to perform on the script step selected from the Edit option pull-down.

Choosing the Modify option will produce the Action dialog box, as shown in Fig. 3, filled with the selected script step's entry field data.

Choosing the Insert option opens the script window line at the selected step. This operation produces the Action dialog box. Also, it causes all the "GOTO's" pointing to that line and subsequent lines to be highlighted. User interaction begins at the selected open line and continues from that point until the user closes/exits the script editing session or the user moves the selection cursor to another step. At the time of move and until a new insertion point is selected, a blank step is opened so that the user may insert new steps into the script.

To perform the Remove option, a non-blank step must be selected before the selection of the Remove option. Selection of the Remove option deletes the selected step from the script and the script window.

Either choosing the Modify, insert or pressing the ENTER key when a blank line step is selected causes the Actions dialog box of Fig. 15 to be displayed. This dialog box allows the user to create a script step activity that is to be created or changed in the current SEF window. There are 19 Actions that are supported by the dialog box as follows:

ACQUIRE

CHANGE PRIORITY

CONVERT

DISCONNECT

GOTO

INTERCEPT

QUEUE

REMOVE

RETURN

START

ACTIVATE FEATURE

COLLECT

DELAY

FLASH

IF GOTO

PLAY

RECORD MESSAGE

RESET

ROUTE

Each of the nineteen Actions has different option selections displayed in the dialog box.

The dialog box shown in Fig. 15 is the default dialog box used to compose an ACQUIRE script step. The user accomplishes this activity by:

1. selecting the ACQUIRE action from the Action drop-down entry list

2. selecting via the Resource group box selection field, the required one of two Resource types for the script step
3. entering the Voice Message Unit identifier in the Voice Message Unit: entry field
4. optionally entering Time Out information in the Time Out: entry field

The pushbutton controls in the Action dialog box operates as follow:

OK: This pushbutton causes the selected Script Step in the SEF window to be changed, and a new blank step is created for additional editing. Also, the edited Step updates the system.

Cancel: This control closes this dialog box and unselect any selected lines, and focus is returned to the main SEF menu bar.

The next menu item in the Script Editing action bar, as shown in Fig. 4, is the Utilities option. From this pull down menu option the user is able to:

1. Activate Script which will replace the current activated script with the latest one that was created or edited.
2. Deactivate Script replaces the current active

script with an earlier version.

3. Activate Override forces an individual application or multiple applications to use a specified script.
4. Deactivate Override forces an individual application or multiple applications to use the conventional application script.

Selecting the "Activate Script" option allows the user to substitute a different script for the current active one. This is accomplished with the dialog box shown in Fig. 16.

This dialog box allows the user to create a script description for the script which is to be actuated. The user accomplishes this activity by:

1. selecting via the Class group box selection field the required one of three classes for the script
2. entering the identifier of the script in the Number: entry field
3. selecting via the Script Step Study group box selection field the required one of the two script step study flags

The pushbutton controls in this dialog box operates as follows:

OK: If the Number does exist in the system, this pushbutton causes the system to activate the desired script.
If the Number does not exist, this pushbutton causes the system to activate the desired script.

Cancel: This control closes this dialog box with no other action.

Choosing the "Deactivate Script" option effectively reverses an "Activate Script" option. When "Deactivate Script" option is selected the dialog box of Fig. 17 is active. This dialog box allows the user to create a script description for the script which is to be activated. The user accomplishes this activity by:

1. selecting via the Class group box selection field, the required one of three script classes for the script
2. entering the identifier of the script in the Number: entry field

The pushbutton controls in this dialog box operates as follows:

OK: If the Number does exist in the system, this pushbutton causes the system to deactivate the

desired script.

If the Number does not exist in the system, this pushbutton causes a warning message which informs the user of the undesirable situation.

Cancel: This control closes this dialog box with no other action.

The "Activate Override" option provides the user with the ability to force the use of a specified script for a particular application or group of applications. This utility is called using the dialog box shown in Fig. 18. This dialog box allows the user to create an application description for the application which is to be overrided. The user accomplishes this activity by:

1. optionally entering the affected application or group of applications in the Application(s): entry field
2. entering the required overriding script identifier in the Override Script: entry field

The pushbutton controls in this dialog box operates as follows:

OK: If the Application(s) and Override script exists in the system, this pushbutton causes the system

to activate the desired override condition. If the Application(s) or override Script does not exist, this pushbutton causes a warning message which informs the user of the undesirable situation.

Cancel: This control closes this dialog box with no other action.

A "Deactivate Override" selection rescinds an "Activate Override." Choosing the "Deactivate Override" option produces the dialog box shown in Fig. 19. This dialog box allows the user to create an application description for the application override condition which is to be rescinded. The user accomplishes this activity by:

1. optionally entering the affected application or group of applications in the Application(s): entry field
2. entering the required overriding script identifier which is to be rescinded in the Override Script: entry field

The pushbutton controls in this dialog box operates as follows:

OK: If the Application(s) and Override Script exists in the system, this pushbutton causes the system to activate the desired override condition. If the Application(s) or Override Script does not exist, this pushbutton causes a warning message which informs the user of the undesirable situation.

Cancel: This control closes this dialog box with no other action.

Thus, in accordance with the present invention, the user may readily program the device through use of scripts or vectors in order to customize the routing of incoming telephone calls in a manner as desired at the site of the user.

The foregoing detailed description has been given for clearness of understanding only, and no unnecessary limitations should be understood therefrom, as modifications will be obvious to those skilled in the art.

What is claimed is:

1. A device for programming script sets in a telephone system, comprising:

means for manipulating the script sets;

means for editing the script sets; and

means for performing tools for the script sets.

2. The device of claim 1 wherein the manipulating means comprises means for creating new scripts.

3. The device of claim 1 wherein the manipulating means comprises means for reading an existing script and displaying the script for modification.

4. The device of claim 1 wherein the manipulating means comprises means for removing an active window and associated dialog boxes from a screen.

5. The device of claim 1 wherein the manipulating means comprises means for reviewing another existing script.

6. The device of claim 1 wherein the manipulating

means comprises means for removing an existing script.

7. The device of claim 1 wherein the manipulating means comprises means for preparing and scheduling a script for printing.

8. The device of claim 1 wherein the manipulating means comprises mean for ending a session of programming.

9. The device of claim 1 wherein the editing means comprises means for changing original script steps.

10. The device of claim 1 wherein the editing means comprises means for creating new script steps.

11. The device of claim 1 wherein the editing means comprises means for deleting original script steps.

12. The device of claim 1 wherein the performing means comprises means for turning on a script.

13. The device of claim 1 wherein the performing means comprises means for turning off a script.

14. The device of claim 1 wherein the performing means comprises means for replacing a script associated with an application.

15. The device of claim 1 wherein the performing means comprises means for rescinding the association of a script with an application.

16. The device of claim 1 including an application script class.

17. The device of claim 1 including a routing script class.

18. The device of claim 1 including an intercept script class.

19. The device of claim 1 including an active script type.

20. The device of claim 1 including a new script type.

21. The device of claim 1 including an archive script type.

Patents Act 1977
Examiner's report to the Comptroller under Section 17
(The Search report)

Application number
GB 9323325.2

Relevant Technical Fields

(i) UK Cl (Ed.M) H4K (KFH, KFD)

(ii) Int Cl (Ed.5) H04M (3/42, 3/50, 3/54) H04Q (3/00, 3/545)

Search Examiner
MR S J L REES

Date of completion of Search
28 JANUARY 1994

Databases (see below)
(i) UK Patent Office collections of GB, EP, WO and US patent specifications.

(ii) ONLINE DATABASES : WPI

Documents considered relevant following a search in respect of Claims :-
1-21

Categories of documents

X: Document indicating lack of novelty or of inventive step

Y: Document indicating lack of inventive step if combined with one or more other documents of the same category.

A: Document indicating technological background and or state of the art.

P: Document published on or after the declared priority date but before the filing date of the present application.

E: Patent document published on or after, but with priority date earlier than, the filing date of the present application.

&: Member of the same patent family; corresponding document.

Category	Identity of document and relevant passages		Relevant to claim(s)
X	WO 92/11603 A1	(BELL) - whole document, only pages 1-59 of description and 1-20 of drawings to applicant	1-21
X	WO 92/11724 A1	(BELL) - whole document	1-21
X	US 4695977	(A.T. & T) - whole document	1-21
X	US 5117372	(A.T. & T) - whole document	1-21

Databases: The UK Patent Office database comprises classified collections of GB, EP, WO and US patent specifications as outlined periodically in the Official Journal (Patents). The on-line databases considered for search are also listed periodically in the Official Journal (Patents).